

KERMIT: logicalization of BERT

YKY

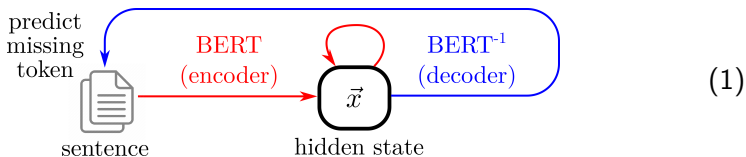
August 2, 2020

Table of contents

- 1 BERT's ground-breaking significance: closed-loop training
- 2 BERT's internal architecture
- 3 Symmetry in logic
- 4 Symmetric neural network
- 5 Logicalization of BERT
- 6 Connection between AI and logic
- 7 What is attention?
- 8 Predicates and propositions
- 9 Attention at the propositional level
- 10 "Attention is all you need" ?
- 11 Content-addressable long-term memory
- 12 Knowledge graphs

BERT's ground-breaking significance: closed-loop training

- BERT uses ordinary text corpora to **induce** knowledge, forming representations that have **universality**:



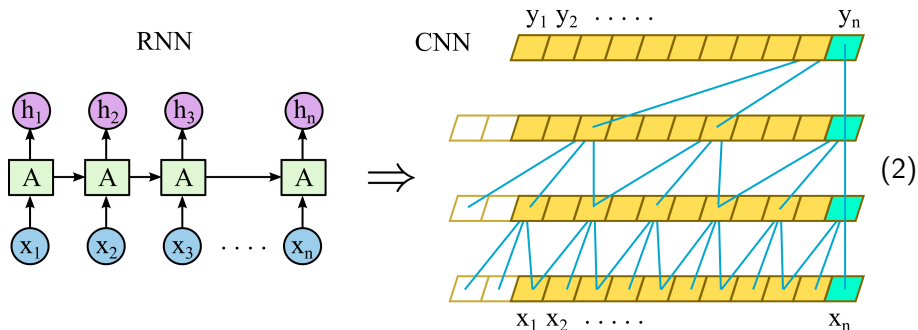
In other words, the hidden state compresses the meaning of sentences, that can be used in other scenarios

- This implies that human-level AI can be *induced* from existing corpora, without the need to **retrace human infant development**
- This training technique came from an earlier paper, unrelated to BERT's internal architecture

BERT's internal architecture

BERT results from combining several ideas:

- BERT is basically a seq-to-seq transformation
- Seq-to-seq was originally solved by RNNs
- But RNNs are slow, researchers proposed to replace them with CNNs



- CNN with **attention mechanism** gives rise to Transformer
- My idea is to incorporate symmetric NN into BERT while following this line of thinking

Symmetry in logic

- Words form sentences, analogous to concepts forming propositions in logic
- From an abstract point of view, logic can be seen as an algebra with 2 operations: a non-commutative multiplication (\cdot , for composition of concepts) and a commutative addition (\wedge , for conjunction of propositions)
- For example:

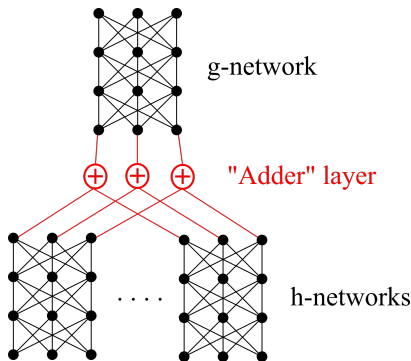
$$\begin{array}{ccc} A \wedge B & \equiv & B \wedge A \\ \text{it's raining} \wedge \text{lovesick} & \equiv & \text{lovesick} \wedge \text{it's raining} \end{array} \quad (3)$$

- Word2Vec was also ground-breaking, but it was easy to go from Word2Vec to Sentence2Vec: just concatenate the vectors
Sentences correspond to propositional logic
- A set of propositions requires symmetric NN to process, as elements of the set are permutation invariant

Symmetric neural network

- The symmetric NN problem has been solved by 2 papers: [PointNet 2017] and [DeepSets 2017]
- Any symmetric function can be represented by the following form (a special case of the Kolmogorov-Arnold representation of functions):

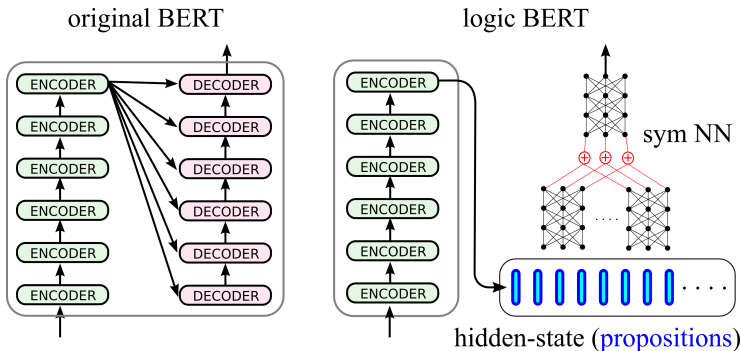
$$f(x, y, \dots) = g(h(x) + h(y) + \dots) \quad (4)$$




(5)

Logicalization of BERT

- We can convert BERT's hidden state into a set of propositions, by replacing the original **decoder** with a sym NN:



(6)

- Permutation invariance of the 's is automatically satisfied by the architecture of the new decoder
- The original **encoder** can be retained, as it is a universal seq-2-seq mapping
- The **decoder** imposes symmetry on the hidden state; Error propagation is expected to cause its representation to change
- Of course, this remains to be proven by experiment 🤔

Connection between AI and logic

- If AI is based on logic, there must exist a **precise** connection between them
- BERT seems to be performing some kind of **transformations** between sentences, such sentences are simply compositions of word-embedding vectors:

$$\text{Socrates} \cdot \text{is} \cdot \text{human} \xrightarrow{BERT} \text{Socrates} \cdot \text{is} \cdot \text{mortal} \quad (7)$$

While this may seem crude, it is effectively the same as a logic formula:

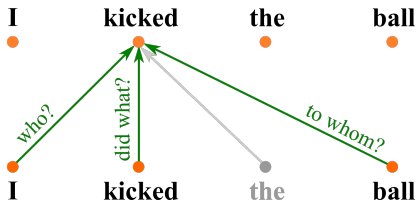
$$\forall x. \text{Human}(x) \rightarrow \text{Mortal}(x) \quad (8)$$

Surprisingly, by the Curry-Howard correspondence, this formula corresponds to the mapping (7) above!

- In another article we shall explore this connection. One could say the mathematical structure of logic is “eternal”; It will provide guidance for the long-term development of AI

What is attention?

- Attention originated with Seq2seq, then BERT introduced self-attention
- The essence of attention is **weighing**
- For each input, attention weighs the **relevance** of every other input and draws information from them accordingly to produce the output
- In BERT, attention is a relation among **words** in a sentence:

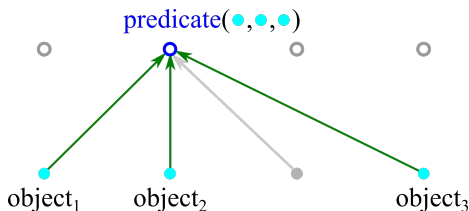


(9)

- From a logic point of view, words \neq propositions
- In logic, the distinction between sub-propositional and propositional levels is of crucial importance!

Predicates and propositions

- The word “predicate” comes from Latin “to declare”
- In logic, a predicate is a declaration without a subject or object; In other words, it is a proposition with “holes”
- **Proposition** = **predicate** + **objects**
- Eg: Human(John), Loves(John, Mary)
- From the logic point of view, the output of attention is the **fusion** of a predicate with its objects:



- Or figuratively:

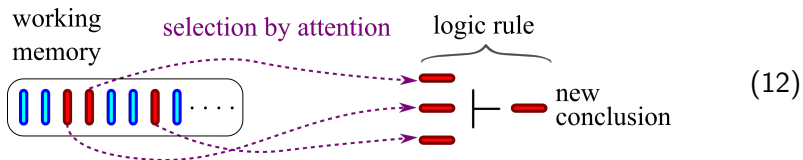
predicate + objects = proposition

$\bullet + \bullet \bullet \bullet \dots = \text{—}$

(11)

Attention at the propositional level

- Analogously, attention on higher levels may process relations among propositions, but here this mechanism seems rather inadequate
- When forming sentences from words, there are relatively few syntactic patterns, such as:
subject · verb · object
- But there are no *a priori* patterns for forming deductions from propositions; Logically related things need not be similar
- Eg: need to pee \wedge not in toilet \Rightarrow hold it
But “pee” and “in toilet” is learned *a posteriori*
- We wish for attention to **select** propositions that are relevant for deduction:



- But to choose N propositions from a set of N , there would be $\binom{M}{N}$ subsets, an **exponential** number

“Attention is all you need” ?

- At the propositional level, attention uses a **weight matrix** to memorize the relatedness among propositions, such a method seems **inefficient**
- Suppose q is the proposition under focus, p_1, p_2, \dots are some possibly relevant propositions, \bowtie denotes matching by attention, we want to output their relatedness:

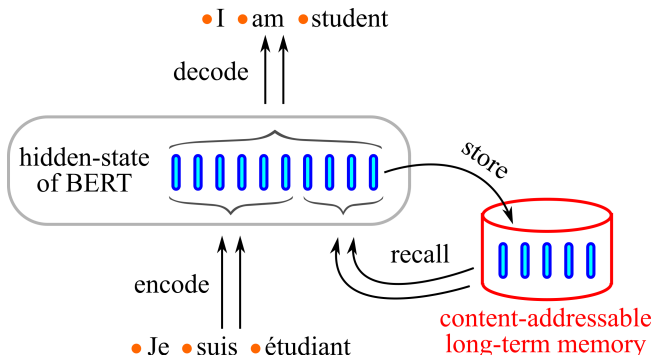
$$q \bowtie p_1, p_2, \dots \mapsto \text{related?} \quad (13)$$

but each case takes one row of matrix for storage

- This is a “flat” representation of cases
- If we try to use **hierarchical** techniques to handle this, it becomes more and more like deep learning, we might as well just use a neural network!
- In other words, use a deep symmetric NN, let it learn **internally** how to select relevant propositions

Content-addressable long-term memory

- The original BERT's hidden state lacked a logical structure; It was not clear what it contains exactly. With logicalization, propositions inside BERT can be stored into a **long-term memory**:



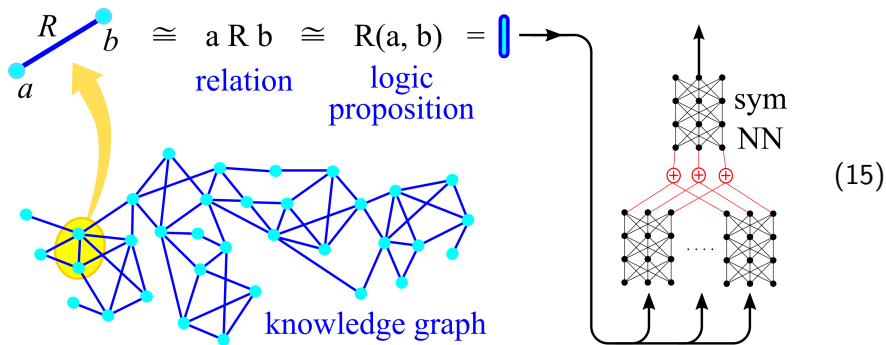
(14)

Eg. "The sun is hot", "Water flows downhill" are facts that stay constant

- Name: **K**nowledge-**E**nhanced **R**easoning with **M**emorized **I**tems
- This is getting very close to strong AI, and depends crucially on logicalization
- The content-addressable memory idea came from Alex Graves *et al*'s Neural Turing Machine [2014]

Knowledge graphs

- One cannot feed a knowledge graph directly into a neural network, as the input must be a vector. A solution is to break the graph into edges, where each edge is equivalent to a **relation** or **proposition**. One could say that graphs are isomorphic to logic



- As edges are invariant under permutations, it seems that we must use symmetric NNs to process them

References

Questions, comments welcome 😊

- [1] Alex Graves, Greg Wayne, and Ivo Danihelka. “Neural Turing Machines”. In: *CoRR* abs/1410.5401 (2014). arXiv: 1410.5401. URL: <http://arxiv.org/abs/1410.5401>.
- [2] Qi et al. “Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation”. In: *CVPR* (2017). <https://arxiv.org/abs/1612.00593>.
- [3] Zaheer et al. “Deep sets”. In: *Advances in Neural Information Processing Systems* 30 (2017), pp. 3391–3401.